

Collaboration with Visual Studio® Team System 2008 Development Edition and Database Edition

White Paper

For the latest information, please see [www.microsoft.com/vstudio](http://www.microsoft.com/teamsystem)

****

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in, or introduced into a retrieval system, or transmitted in any form, by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2007 Microsoft Corporation. All rights reserved.

Microsoft, Excel, MSDN, SharePoint, SQL Server, Visual SourceSafe, Visual Studio, and the Visual Studio logo are trademarks of the Microsoft group of companies.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Contents

Introduction 1

Collaborate and Communicate 2

Drive Predictability 3

Manage Team Workflow 4

Use Familiar Tools 5

Ensure Quality, Early and Often 6

Integrate Work Frequently 7

Make Real-time Decisions 8

Summary 9

# Introduction

Writing software is hard.

Writing software as a team is harder.

The vision of the lone developer cranking out brilliant applications in a basement or cubicle, although romantic, is rapidly becoming a thing of the past. With few exceptions, today’s business applications are too broad, too large and too complex for any one developer to write alone. It takes a team to develop most business software. And on any team, they need effective collaboration.

Visual Studio® Team System 2008 Development Edition and Visual Studio® Team System 2008 Database Editionare two of Microsoft’s latest versions of their powerful Visual Studio development environment. Both provide potent tools for creating .NET code and database code. Both provide powerful tools to help improve code. And both provide the tools necessary for effective team collaboration.

# Version Control

Let’s start with the most basic collaboration information possible for team software development – version control. Through their integration with Visual Studio® Team System 2008 Team Foundation Server, both Visual Studio Team System 2008 Development Edition and Visual Studio Team System 2008 Database Edition provide professional version control of both application code and database code (including schema information). If you’re thinking “How is version control a collaboration tool?” you’re completely justified. Historically, the majority of version control tools provide very little in the way of collaborative information.

Visual Studio Team System 2008 Team Foundation Server provides a great deal of information to team members simply from the way it implements version control. In Visual Studio Team System 2008, checking out a file isn’t a one dimensional operation—it provides valuable meta-data as well. For instance, who checked out the file, when they checked out the file, and how they checked out the file is all available to other team members. So depending on what you’re trying to accomplish, you’ll check out files in different ways. If you’re working on adding a small piece of functionality to a class, you’ll check out the file without locks. Other developers can see that you have the file checked out, but that you didn’t lock it. They will feel confident in checking out the code as well, knowing that any merge conflicts during their check-in will likely be very manageable. On the other hand, if you plan to do a major refactor of a class, you’ll lock the code for both check-out and check-in, alerting your team members that they should wait until you check in the radical changes before they try to change some minor functionality.

Traditionally, a lot of database developers didn’t keep their databases under version control at all. And, if it they did, they only protected the stored procedure code, not the entire schema. However, Visual Studio Team System 2008 Database Edition changes the game by allowing you to track changes to your database objects. By keeping the database schema and code under version control, database developers can far more effectively collaborate on incremental changes to the database schema and code, and can understand the changes made by other developers. For more information about keeping database objects under source control, see the white paper, “Database Change Management” or consult the Visual Studio Team System 2008 Database Edition product documentation.

Although subtle, version control as a collaboration tool is surprisingly powerful. It also highlights a characteristic of the technical team members: their collaboration is both unobtrusive and low key. Using version control in this way, team members don’t need to send e-mail announcements, generate paperwork, attend meetings or restrict each other’s process. Simply by doing their jobs right they can communicate, elegantly and efficiently.

# Code Annotation

Other technical, collaboration features provided by Visual Studio Team System 2008 and exposed in the Visual Studio Team System 2008 Development Edition and Visual Studio Team System 2008 Database Edition are nearly as clever and discreet as version control. One of these features, Annotate, allows a team member to look at any code file (application or database), and see the complete history of the current code lines in one view. Each line of code is annotated with the name of the developer that wrote that line of code, the date and time it was written and, most importantly, a link to the other code files that were checked in at the same time.

The collaborative power of this feature is hidden to most non-technical users. However, a developer will often immediately recognize the value. As an example, imagine you’re working on fixing a difficult bug in some legacy code. As you narrow the problem down to a few lines of code in one file, you can immediately *annotate* that code and discover when and by whom the line of code was written. More importantly, you have an immediate link to the other files that developer was working on when they introduced the bug. There is a strong chance the same bug, or related problems, might also exist in those files. This collaboration feature is subtle, but it is very powerful.

# Collaboration through Code Analysis

Another powerful, yet discreet, collaborative feature is code analysis. This feature allows you to look at several metrics on your code. Visual Studio Team System 2008 Development Edition features several tools to help you measure code metrics, all of which help understand the quality of the code you’re creating. Depth of Inheritance will show you how deep your inheritance hierarchies go. Class Coupling shows the dependencies between your classes and how loosely coupled your code is. Cyclomatic Complexity can show the complexity of individual methods, and can tell you if each method is trying to accomplish too much. The Lines of Code metric, although not usable as a productivity metric, is very useful for determining the overall size of a piece of functionality. And finally, the Maintainability Index gives you an idea on how maintainable your code is.

Here is the collaboration angle: code development is a team effort, and by focusing on keeping your code maintainable, and your code metrics in good shape, you provide maintainable code that can improve the productivity of the entire team. For more information about using code metrics to improve the quality of your code, see the white paper “Using Visual Studio Team System 2008 Development Edition and Visual Studio Team System 2008 Database Edition to Improve Code Quality” or see the Visual Studio Team System 2008 Development Edition product documentation.

# Code Reviews

Not all collaboration between technical team members is inconspicuous. Code reviews are an observable collaboration tool. And the Visual Studio Team System 2008 Development Edition and Visual Studio Team System 2008 Database Editions have tools to help simplify code reviews. A feature called code shelving allows developers to share their code changes, even across multiple code files, with other developers. This facilitates code reviews by allowing simple sharing of all the changed source code files to implement a particular feature or bug fix. Here’s how it works. One of your coworkers fixes a performance bug, and shelves his proposed solution. You pull down your coworker’s workspace (which, through the magic of Visual Studio Team System 2008 version control, contains all of the right versions of all of the files that your coworker used to create his code). Then you grab the shelveset from the server, pulling it into the compare tool so that you can see the highlighted changes between the original code and the bug fix. After reviewing the code and approving the changes, you can check in the code on behalf of the original developer, or let them know that the code is ready to be checked into version control.

# Using Work Items to Promote Collaboration

Not all collaboration is with the technical members of your team. You also collaborate effectively with the non-technical members of your team, and the Visual Studio Team System 2008 tools available in the Visual Studio Team System 2008 Development Edition and Visual Studio Team System 2008 Database Edition can help. One of the most prominent means of collaboration in any Visual Studio Team System 2008 implementation is through work items. Work items are tracked units of work that can be assigned to individuals or teams, and that follow a configurable workflow. You can also use work items as key components of an effective collaboration strategy. Because work items are used by both technical and non-technical team members, they provide explicit information that is generally in text. This enhances the communication between people with different levels of technical expertise since it requires less assumed knowledge.

One of the most important aspects of software development is the effective gathering and dissemination of software requirements. However, one of the main difficulties with modern software development is that requirements tend to change rapidly, even during the actual software implementation. Likely this has happened to you. In fact, as a developer you may have been in the actual coding phase of a requirement when it was changed. Given your desire to be responsive to the business you might not be able to stop those changes from occurring, however, you want to make very sure that you’re notified of that change! Many a developer has been frustrated by developing against an old version of the requirement. You can minimize this frustration if your team keeps their requirements in Team System. If the team stores the requirements as work items, each member on the team can choose to sign up to be notified, via e-mail if necessary, whenever the requirement you’re working on changes. Also, by having a single repository, you can be assured that the version of the requirement you’re coding is the latest.

Work items have a more profound effect on team collaboration than through requirements alone. Since Visual Studio Team System 2008 can track so many different types of work items, they can become a tool for measuring status and progress through a development process. This touches on reporting as a communication tool for any software development team. Depending on the organization, status reporting can be as simple as letting people know what you’re working on, to as complicated as manually tracking and reporting time spent on individual tasks. When developing using the Visual Studio Team System 2008 Development Edition or Visual Studio Team System 2008 Database Edition, you track most of your status by simply fixing bugs, implementing features and checking in code–in other words, doing your job. Rather than repeatedly asking you for updates and status, project managers have their own selection of tools and reports that allow them to track the progress of the development. That means the weekly status report your manager expects can be created without needing your direct input! In addition, by using work items, your team doesn’t depend on e-mail messages and meetings that drain productivity and result in lost data. Decisions communicated in e-mail messages tend to get lost over time, while those stored in work items are persisted and available to the entire team. Decisions made in meetings also tend to get lost, unless accurate meeting notes are available in a centralized location, which you have with Visual Studio Team System 2008.

Overall, the Visual Studio Team System 2008 tools available in both the Visual Studio Team System 2008 Development Edition and Visual Studio Team System 2008 Database Edition provide powerful collaboration capabilities, for both technical and non-technical collaboration. Additionally, the tools are geared appropriately to the needs of the users and their general styles of communication. Technical collaboration tools tend to be very unobtrusive, and rely on shared technical understanding between team members, while non-technical collaboration tools tend to be more explicit. For a successful project, both forms of collaboration are required. And Visual Studio Team System 2008 supports you with the tools you need for both technical and non-technical modes of collaboration.

For more information, visit the Web site at <http://www.microsoft.com/defyallchallenges/teamsystem>.